

# Flexible classification standards for product data exchange

Wolfgang Wilkes<sup>1</sup>, Peter J. A. Reusch<sup>2</sup>, Laura Esmeralda Garcia Moreno<sup>2</sup>

<sup>1</sup> Fernuniversität Hagen, Germany, <sup>2</sup> Fachhochschule Dortmund, Germany

## ABSTRACT

One essential part of e-business is the exchange of product data between business partners. Classifications have been developed as a means to clearly describe the semantics of product descriptions. They provide schema elements like properties and classes and define their semantics by some formal relationships and some textual (informal) definitions. This chapter gives an overview about the modelling levels which have to be considered for the development and use of classifications. In addition, it introduces briefly ISO 13584 (PLIB) as one important data model for classifications and it characterizes a number of classifications which are used in today's product exchange processes. Many of the classification standards use a quite primitive data model which leads to problems in their use and maintenance. By exploiting some features of more advanced data models, many of these problems can be overcome. We propose the introduction of additional class hierarchies as an example for adding more flexibility to a classification and discuss this proposal in the context of eCl@ss, an important European classification.

## INTRODUCTION

A fundamental requirement for implementing full e-business is the ability to exchange product information between different business partners and their software systems. As far as product-numbers, prices, delivery information, etc. are concerned, exchange is possible on the basis of general models which are applicable for any kind of product. But the technical diversity of products leads to a diversity of technical descriptions. This makes it impossible to define a general model covering all aspects of all types of products in a concise way: The description of e.g. bolts and nuts requires fundamentally different information than the description of integrated circuits or refrigerators.

Whereas STEP, a series of standards on product modeling (International Standardization Organisation [ISO], 1994), defines a big number of models for various domains to describe e.g. geometry or other representation models of a product, the exchange of technical product information in e-business and e-engineering is basically done by describing products by their characteristics or properties. This information exchange is normally based on a meta data approach: Information about a property of a product is exchanged as a pair (property\_ref, property\_value), where the property\_ref is an identifier of a concept in a classification (often also called a dictionary or product ontology). For the correct interpretation of the property\_value, the receiving system has to refer to the classification, where the meaning of the property is defined (textually and possibly supported by graphical means) and further information is available like names (in different languages), synonyms, data type, unit of measure, relationship to other

concepts, etc. Thus, we have a very simple structure for the exchange, but we need classifications as additional resources which are referenced from the exchange structure.

The consistency and accuracy of the elements of these classifications is important for users: If the resources of a classification standard are not able to describe their products appropriately they will not be able to transmit the required information and to use the classification standards in their product data exchange processes.

Most classification standards are developed and maintained by a consensual process which is based on industrial working groups which consider the change requests from users and decide on their integration into the existing classification standard. The difficulties of these activities should not be underestimated, in particular in view of the growth of the classification systems over time. Today, classification standards often consist of more than 10,000 classes, several thousands of properties, and an even bigger number of class-property relations. Due to the increase of product areas which are being integrated in product classifications and the increase of requirements of applications, the growth of classification standards is enormous. For example, Table 1 shows the development of the number of classes and properties over the last versions of eCI@ss (figures obtained in personal communications with members of the eCI@ss-office, some information can also be obtained from CEN, 2010).

	<b>V4.0</b>	<b>V4.1</b>	<b>V5.0</b>	<b>V5.1</b>	<b>V5.1.1</b>	<b>V6.0</b>	<b>V7.0</b>
<b>Publication Date</b>	2001/08	2002/09	2003/12	2004/09	2005/09	2008/04	2011/02
<b>Commodity Classes</b>	10,190	12,565	20,379	21,100	22,203	32,590	37,868
<b>Properties</b>	2,303	5,504	3,667	5,525	6,941	10,930	15,397

*Table 1: Growth of Classes and Properties in eCI@ss*

Thus, current classification standards have to face a number of problems:

- The classification standards are growing, so that the organization of the maintenance process becomes more difficult.
- Different user groups put different requirements on the classification standards.
- Classification standards are based on simple data models which put limitations on the expressiveness and flexibility of classification standards.

Based on an overview of classification models and standards, we will argue in this chapter that future classification standards have to provide different views on a common set of basic classes and properties. This will allow for a better support of conflicting requirements, and it will also support the maintainability of big classification standards. Basis for such an improvement of classification standards is the use of data models which provide sufficient resources to build a more flexible classification standard.

Our example is eCI@ss, a horizontal classification standard with the approach to cover all industrial products in all phases of their lifecycle (see [www.eclass.eu](http://www.eclass.eu)). We will illustrate for eCI@ss that by means of the capabilities of its new data model multiple classification hierarchies can co-exist. This will result in a number of benefits for its maintenance and use, both by

reducing the number of class-property relationships and by increasing the consistency of the classification standard.

The rest of this chapter is organized as follows: In section 2 we give an overview about classification standards and their capabilities to describe and organize product information. We introduce the standard ISO 13584-42 (ISO, 2010), also called PLIB, as a data model which is increasingly used for classification standards and show how eCl@ss has adopted this data model in its recent version 7.0. In section 3, we highlight some of the problems which result from deficiencies of the simplistic data models. Section 4 describes the definition of additional class hierarchies for a classification to overcome these problems, and it contains some calculations for our example eCl@ss about the reductions of property-class assignments. Finally, in section 5 we give a summary and an outlook on further work.

## **CLASSIFICATION STANDARDS AND THEIR CAPABILITIES**

### **Fundamentals of classification standards**

For describing product data which can be exchanged across different partners and their systems, the meaning and organization of data in exchange files or messages needs to be specified. This is done by defining a model or a schema to which the objects in the exchange file can be related to describe their semantics. Such models are called classifications in this chapter. Unfortunately, this terminology is not very clear and in different contexts a variety of names can be found having a similar meaning. Examples of such models, which also give an impression about the variety of terms, are

- E-business classification standards like eCl@ss ([www.eclass.eu](http://www.eclass.eu)), ETIM ([www.etim.org](http://www.etim.org)), Proficl@ss ([www.proficlass.org](http://www.proficlass.org)), and UNSPSC ([www.unspsc.org](http://www.unspsc.org)),
- ISO and IEC standards providing product dictionaries like ISO 13584-511 (ISO, 2006) about fasteners, ISO 23584 (ISO 2009) about optical devices, and IEC 61987 (IEC, 2009b) about control instruments,
- Reference libraries like ISO 15926-4 (ISO 2007),
- RosettaNet Technical Dictionary (RNTD) for electronic components (RosettaNet, 2007).

Classifications specify sets of concepts which are required to describe the products of a domain or of several domains. For instance, ISO 13584-511 (ISO, 2006) contains various classes of fasteners together with their properties which can be used to describe a specific fastener in a catalogue.

As a matter of fact, these classifications provide different kinds of information about the products of their domain. Some examples:

- ISO 13584-511 (ISO, 2006), ISO 23584 (ISO, 2009) and IEC 61360 (IEC, 2009a) organize their product classes in a hierarchy which is driven by property inheritance, i.e. for upper classes in the hierarchy, properties are defined which are inherited by subclasses.
- Other classifications do not provide a class hierarchy, e.g. NE 100 (International User Association for Automation in Process Industries [NAMUR]), 2010), OTD (ISO, 2010b), and RNTD (RosettaNet, 2007).
- Sometimes, a hierarchy of classes is provided which is not based on property inheritance but on market segments (eCl@ss, UNSPSC, the Global Product Classification of GS1

(GPC, see <http://www.gs1.org/gdsn/gpc>), and the Common Procurement Vocabulary (CPV, European Commission, 2008).

- ISO 15926-4 (ISO 2007b) only defines the classes and properties but does not relate them to each other.

Thus, the classifications provide different facets or features as a means to actually describe products in an exchange (or in a database). Which features they provide is based on two (not independent) elements:

1. The intention of the provider: For what purposes is the classification supposed to be used?
2. The capabilities of the underlying data model which provides the means for building classification.

Based on their intentions, the classification providers will decide whether they need a class hierarchy, how the class hierarchy is organized, how classes and properties are related, etc. But if the providers have selected a data model they are bound to the features which are supported by this data model. Examples of data models for classifications are ISO 13584-42 (ISO 2010a, will be described in more details below), ISO 15926-2 (ISO, 2003), ISO 12006-3 (ISO, 2007a, also known as IFD, International Framework for Dictionaries), and ISO 22745 (ISO, 2010b, OTD). And many classifications use variants thereof or use their own data models.

From a modeling point of view, we have to distinguish different modeling levels, the data model, the classification and the product description itself:

- Level 1: data models, providing the capabilities and methods for describing concepts and for providing means to deal with them.
- Level 2: classifications, providing a set of concepts to describe products
- Level 3: Instances in a database or a catalogue standing for products or components which are described following the specifications of a classification.

In the following example, these levels are illustrated for screws:

- Level 1: Modeling elements like class and property and their relationships are defined in the data model (e.g. PLIB).
- Level 2: Concepts like the class *Screw* and the related properties *Form\_of\_head*, *Screw\_length* and *Length\_of\_thread* are elements of a classification. They normally are identified by unique concept identifiers.
- Level 3: A screw in the catalogue of the manufacturer SCREW Ltd. with product-ID 12345 might have the following property values:

*Form\_of\_head* = 'hexagonal head',  
*Screw\_length* = 6 cm.  
*Length\_of\_thread* = 4 cm.

- There might be a fourth level: The specific screw with product-ID 12345 which is used in my desk to fasten the right front leg. This screw might be identified by a serial number (which is not very likely in case of screws).

## ISO 13584 (PLIB): A data model for classifications

ISO 13584 (ISO, 2010a) is a standard which provides a formal data model for classifications. The model is defined using the modeling language EXPRESS (ISO 10303-11) (ISO, 2004). The two responsible committees for this standard are ISO TC 184 SC4 WG2 and IEC SC3D, so that the same model is also used in IEC 61360 for the description of electro-technical components.

The most important concepts which are provided by PLIB are classes and properties. Classes represent product groups and product types (which might be abstract), and properties specify the characteristics of products.

PLIB distinguishes between two fundamental types of classes:

- Categorization classes are used to group products into sets. An example is the UNSPSC code system. Categorization classes cannot have properties.
- Characterization classes provide properties which can be used to characterize products.

Class hierarchies can be built by two types of relationships:

- The *is-a* relationship builds a strict tree (or a set of trees) of characterization classes. *is-a* includes complete property inheritance (i.e. each property of the higher level class is inherited by the lower level classes).
- The *case-of* relationship relates any kind of classes (characterization and categorization classes), and it allows an individual inheritance of properties. The relationship defines which properties are "imported" into the lower level class or which properties of the lower level class can be unified with properties of the higher level class. One application is the connection of one classification to another one where a class in the first classification is related to a class in the second classification with the purpose to import some of the properties of this class into the first classification.

Properties have always a definition class, i.e. they are defined for that class and all subclasses thereof. But they may not be used automatically for characterizing products of a class – for that purpose, the property has to be made "applicable" for this specific class. This distinction between the class for which a property is defined and the applicability of that property allows for an exception mechanism: some subclasses of a superclass may allow characterizing their products with an inherited property; other subclasses may not allow this.

The type of a property defines the allowed values for describing a product. Examples are simple types like integer, number, string, or more complex types like measured types (which have a unit), aggregate types (arrays, lists and sets), and references to other classes which are used to establish relationships between products with the meaning of composition (i.e. the referenced instance is regarded as a component of the referencing instance).

PLIB allows organizing properties according to a specific point of view. The general properties describe the product as it is. If these values are changed, normally the product really changes. But there exist other information about products which describe other aspects of them. For instance, the prize information may change over time without changing the product itself. PLIB provides a mechanism to organize properties in aspects which describe different views of the product.

Both classes and properties can be described by a number of human readable information: preferred name, synonyms, definitions, notes, remarks, graphical elements, etc. Most information can be presented in several languages (preferred name, definition, notes, remarks, etc.), and synonyms may also be of different languages.

## **Brief overview about some classification standards used in e-business**

In this subsection, we give a brief overview about some of the important classification standards which are referred to in e-business processes for exchanging product information. We give a short characteristic of the classifications, their domains of use, and their capabilities in terms of the used data model.

### **UNSPSC (United Nations Standard Products and Services Code)**

UNSPSC (see [www.unspsc.org](http://www.unspsc.org)) is a classification which is owned by the United Nations and which is maintained by GS1 US. It is a horizontal, branch spanning classification which is mainly used in the US, the UK and the Scandinavian countries. But it is also in use in many other countries, and it is probably the most used classification standard worldwide.

UNSPSC is a pure classification standard in the sense that it only provides a hierarchy of classes to assign products to a specific product group. It does not contain properties to describe products in more detail. The class hierarchy is built to collect products into classes which can, for example, be put together on a shelf, in a rack, or in a complete store. One example of high level UNSPC classes highlights this:

*UNSPSC Code 60 00 00 00:*

Musical Instruments and Games and Toys and Arts and Crafts and Educational  
Equipment and Materials and Accessories and Supplies

*UNSPSC Code 60 10 00 00*

Developmental and professional teaching aids and materials and accessories and  
supplies

From a data model point of view, UNSPSC provides classes for categorizing products which are linked in a hierarchy and just give the class name, a code, and a definition of the class. The class hierarchy is fixed to four levels, and the UNSPSC code describes the position of a class in the hierarchy (60 00 00 00 is a top level element, 60 10 00 00 is at the second level in the hierarchy, etc.).

### **GPC (Global product classification)**

The GPC (<http://www.gs1.org/gdsn/gpc>) is a classification which has been developed by GS1, a worldwide organization providing standards and an infrastructure for the exchange of product information in e-business transactions. GPC is part of this infrastructure called Global Data Synchronization Network (GDSN). As such, the GPC is used by a big number of users who are related with this infrastructure.

The GPC is also a 4 level classification, and the classes at the lowest level are called bricks. Bricks may have attributes, and the purpose of these attributes is not to describe products but to define a variant of the brick. With the selection of values for the attributes, a brick variant is selected. Example: the brick *Printed Books/Compositions* has an attribute *format* with possible

values like *hardcover/hardback*, *paperback/softback*, *loose leaf*, etc. By selecting one of these values, a specific brick variant is selected.

## ETIM (electro technical information model)

ETIM ([www.etim.org](http://www.etim.org)) is a classification which is specialized for the description of electro-technical products. It has been developed and is supported mainly by retailers of these kinds of products. National ETIM associations have been established in a number of European countries, e.g. Germany, the Netherlands, Belgium, and Austria.

ETIM emphasizes the description of products by means of properties. As such, ETIM does not provide a complex hierarchy, it only has a two level hierarchy of product groups, and the properties are associated with the classes of the second level. A specific feature of ETIM is the class specific definition of predefined value sets for alphanumerical properties: The allowed values for a property depend on the class to which the product is associated.

## eCl@ss

eCl@ss ([www.eclass.eu](http://www.eclass.eu)) is a horizontal classification standard including products of various industrial sectors. It is used by industry and trade companies, having started in Germany around the year 2000, and since then experiencing an increasing use in Europe and worldwide. It is driven by an industrial association based in Germany.

Similar to UNSPSC, eCl@ss provides a 4 level class hierarchy, and similar to UNSPSC, to each class an eight-digit code is attached (the eCl@ss code) which identifies the class by its position in the hierarchy. Different to UNSPSC, eCl@ss defines properties for the description of products. These properties are not class-specific (as is the case in ETIM), i.e. a property is used without adaptation across all classes to which it is assigned. The properties are only added to the fourth level of the class hierarchy, i.e., the hierarchy is not built on the concept of property inheritance.

With the new version of eCl@ss (version 7.0 of February 2011), eCl@ss has adopted PLIB (ISO, 2010a) as its underlying data model. This will be explained below in more detail.

## NE 100 (Namur Recommendation), IEC 61987

NE 100 (Namur, 2010, Löffelmann, Polke & Zgorzelski, 2007) is a standard for the description of electrical devices in the process industry like measuring instruments, pumps, etc. NE 100 has been developed by the Namur working group Prolist which has been further developed to an independent organization (Prolist International, [www.prolist.com](http://www.prolist.com)). The Prolist standard is being converted to a formal standard under IEC 61987 (IEC 2010b).

The NE 100 has been developed to support a clearly defined process: It is supposed to be used for the communication of designers of process plants with the manufacturers of devices used in these plants. Thus, the classification is used mainly in engineering which results in much more detailed product descriptions (some product classes contain more than 1000 properties) and the use of complex structures for defining these properties. Whereas most of the previously described classifications only use simple data types for their properties, NE 100 uses composite properties called blocks which may be of different types (polymorphism) and allows to associate multiple numbers of these blocks to a product. For example, a measuring instrument may incorporate an unspecified number of different measuring functions (temperature, flow rate, etc.). NE 100 does

not provide a classification hierarchy for products, but it contains a hierarchy of blocks which is based on the *is-a* relationship and provides the basis for the polymorphic references to block-type properties.

NE 100 is based on the PLIB data model, and it is probably the most advanced PLIB classification in terms of exploitation of features of this data model. Prolist will merge with eCI@ss in 2012 or 2013, and this has been one of the drivers for eCI@ss to adopt PLIB as their underlying data model.

## **Adoption of PLIB as the underlying data model of eCI@ss**

The original data model of eCI@ss provided four levels of classes: segments, main groups, groups and commodity classes. Properties were only linked to the commodity classes.

To map this model to PLIB, it is important to understand that the class hierarchy is not a *is-a* hierarchy where types of products are specialized from top to bottom (e.g. a general screw is specialized to a hexagonal screw). Rather, the purpose of the eCI@ss hierarchy is to support applications like spent analysis instead of product refinement. This has been pointed out by Hepp, 2006 who revealed that a number of mappings of eCI@ss and UNSPSC to OWL are semantically incorrect: They represent the class hierarchy with an *rdfs: subclassOf* construct which leads to the wrong semantics regarding the subclass as a specialized concept of its superclass. This would, for instance, lead to the wrong implication that a screw assortment box (eCI@ss code 23-11-92-04) is a specialization of a Screw / Nut (eCI@ss code 23-11-00-00).

Thus, the eCI@ss classes must not be related by the *is-a* relationship. PLIB provides the construct of categorization classes which have been specifically introduced to model classes of eCI@ss and UNSPSC, and they are not related by the *is-a*, but by the *case-of* relationship. Categorization classes do not possess any properties, thus the properties cannot be linked as before to the classes at the lowest level. Rather, a new kind of class has been introduced in eCI@ss 7.0: application classes which correspond to the characterization classes of PLIB. Currently, an eCI@ss application class carries the properties of one commodity class to which it is related by a *case-of* relationship, and the application classes are not organized in any form of class hierarchy (see Hesselmann and Ondracek, 2006).

## **PROBLEMS RESULTING FROM THE SINGLE CATEGORIZATION HIERARCHY**

As has been shown in various publications (Hepp, Leukel & Schmitz, 2007, Reusch & Garcia, 2008, Reusch & Garcia, 2009), the horizontal classifications are dealing with a growing number of classes describing similar product categories in different industry domains. For example, many eCI@ss classes describe different kinds of screws, e.g. cortical screw, spongiosa screw, special screw, wood screw, etc. Due to the organizational structures and the capabilities of the underlying data model, these classes cannot be automatically related and thus analyzed for common properties, etc. Normally, the top level branches of many industry spanning classification standards (like eCI@ss and UNSPSC) represent different industrial sectors, thus the working groups are responsible for separated branches of the class hierarchy. And the simple data models used for these classifications does not support a harmonized organization of identical properties.

With the old data model, eCI@ss has only one mechanism for modelling (at least) two different situations: On the one hand, eCI@ss provides a categorization hierarchy which models markets of



products. Therefore, the class hierarchy is organised according to product domains which represent specific industrial branches. The goal is to give companies a clear part of the hierarchy where they can find the relevant products of their industrial sector. Examples are segment 34 (Medicine, medical technology) and segment 23 (Machine elements) where products of these domains should find their home (see Figure 1). But there are also products which exist in various domains. For instance, we find in many segments commodity classes which represent different kinds of screws. Of course, these screws may have different properties, e.g. a medical screw might need to have a certificate related to its allergic potential, but there are a number of properties which are common for all screws across the segments.

The problem is that this cannot be supported and enforced due to the limitations of the used data model. No means exist to provide the information that different classes represent products of a similar kind and might share a number of properties. Therefore, properties need to be assigned independently to all the similar classes – which is a tedious work, but worse, is a potential source of inconsistencies and incorrect definitions.

In the following, we show how this can be overcome by using the features of the PLIB data model, and how both views can be modelled explicitly – the domain specific view with different domain specific classes and the product centric view which relates similar classes in different branches of the hierarchy.

## **SUPPORTING DIFFERENT VIEWS ON A CLASSIFICATION**

### **Building an application class hierarchy**

If we consider the features of the PLIB data model, then eCI@ss used to build its hierarchy only by use of categorization classes. But what we want to achieve is a relationship between classes which would allow the sharing of common properties. Currently, there does not exist any relationship between similar classes in different branches, like screws in the segment of machine element devices (e.g. 23-11-01-01 Hexagon head cap screw) and screws in the segment of medical devices (e.g. 34-32-17-01 Cortical Screw), and it is not possible to share properties across these classes in a controlled way.

With the new concept of application classes the properties are moved out of the hierarchy. They are organised in a separate set of classes, the application classes. Because for E-Business standards continuity is essential we have developed the following solution: We keep the current eCI@ss classification hierarchy, but add a second *is-a* hierarchy of application classes with property inheritance. This is possible according to the PLIB model where we can define many categorization hierarchies alongside one *is-a* hierarchy.

In this model, the application class containing the properties of the commodity class has two ancestors: one ancestor is linked by *case-of* (the commodity class) and the other one linked by *is-a*. This second ancestor will define properties which are used by several similar product groups and which are inherited by the application class. By putting these common properties into higher level application classes, the specific application classes, which, for instance, represent the two screw types mentioned above, need to define only properties specific for this level of the product class, whereas the common properties are inherited from the ancestor.

This is traditional object oriented modelling– what is new in our approach is the combination of two hierarchies giving two viewpoints on the set of classes:

- The original categorization class hierarchy allows the building of branches representing industry domains, and

- The new application class hierarchy allows to factor out common properties of similar classes into a more abstract class thus providing a relationship between classes for sharing common properties.

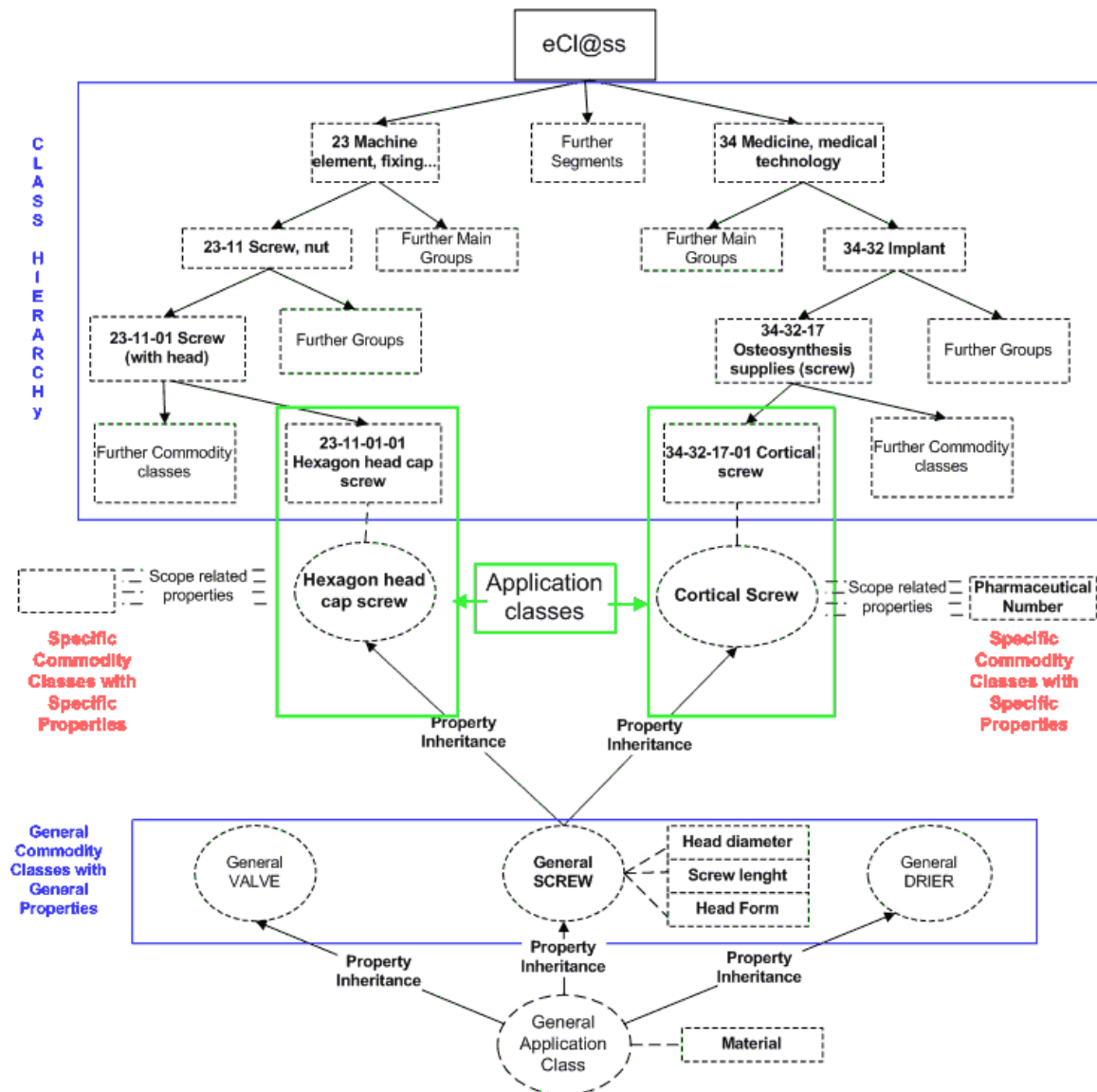


Figure 1: eCl@ss hierarchies with application classes

Figure 1 illustrates the addition of the inheritance hierarchy to the existing classification structure. The upper part of the picture shows the existing eCl@ss hierarchy, whereas the bottom part shows the new inheritance hierarchy of the application classes. There exists a 1-1 relationship between a leaf application class (e.g. Cortical Screw) and a commodity class (e.g. 34-32-17-01 Cortical Screw).

In our example, we use three levels of application classes: a General Application Class Level, a General Commodity Class Level with general properties for a specific high level kind of product (like screws) and Specific Commodity Classes with specific properties for the specific commodities. Properties are inherited down the hierarchy (up in the picture). For example the

property ‘Material’ which is defined for the General Application Class is inherited by the commodity class Cortical Screw, and in the same way, it is a property of other classes like ‘Valve’ or ‘Drier’ etc.

Properties like ‘Head diameter’, ‘Screw length’ or ‘Head form’ are defined for the General ‘Screw’ application class and are inherited, for instance, by the ‘Cortical Screw’ application class. The ‘Cortical Screw’ application class is specific by its additional properties like ‘Pharmaceutical Number’ and other medicine specific properties.

This organisation of properties by inheritance reduces drastically the number of explicit assignments of properties to classes. In addition, it makes obvious that classes in different branches of the eCI@ss hierarchy are of the same kind and share some properties. Thus, we can firstly ensure that properties in these classes are harmonised and will be used with a common semantics, and secondly we can assure that a minimal set of properties is applicable to all similar classes (e.g. the length of a screw to all screw classes). In addition, it is possible and easy to add additional generic properties to all similar commodity classes (by only adding the property to a generic application class).

There exist two mechanisms in the PLIB data model to “filter” the properties which are inherited by the next level:

- PLIB distinguishes between the domain of a property and its application. Each property has exactly one characterization class as its domain. As a result, this property is *visible* in this class and its subclasses. But to use this property actually for describing products, the property needs to become *applicable* for that class. A property may become *applicable* for the domain class itself, but it may become *applicable* also only in some of the subclasses of the domain class. Thus, if the domain of a property is a generic class, it can be specified in the subclasses if this property is *applicable* for the subclass. If a property is *applicable* to a class, then this holds also for all the subclasses (i.e., the status *applicable* is inherited by all the subclasses). As a result, it is possible to prohibit the use of a property for describing products of a specific class even if the property has been defined (as *visible*) for the class or its superclass. The property can only be used in product descriptions if it has been made *applicable*.
- PLIB provides two “specialization” relationships between classes: The *is-a* relationship is the usual inheritance relationship known from object oriented modeling (with the extension of *visible* and *applicable*). In addition, it provides the *case-of* relationship. This relationship is basically intended to allow a linkage between classes of different dictionaries (e.g. to link a company specific product class to an eCI@ss class, where some properties of eCI@ss are imported and some additional company specific properties are added in the product class). This relationship allows the selective import of properties from the general class to the more specific class – thus it allows filtering out not relevant properties and it does not require the inheritance of the complete set of properties by the subclass.

Of course, our three layer approach for the application class hierarchy may be changed according to the requirements of the organisation of properties. For instance, with the separation of physical products and services, we could introduce an additional general application class layer. Services have typical properties like duration, object of provision, personnel qualification, etc. Such properties are candidates for a general application class for services. Then the general application class we used in the previous section becomes the general application class for industry. On top we can introduce another level of application class linking these two branches together and

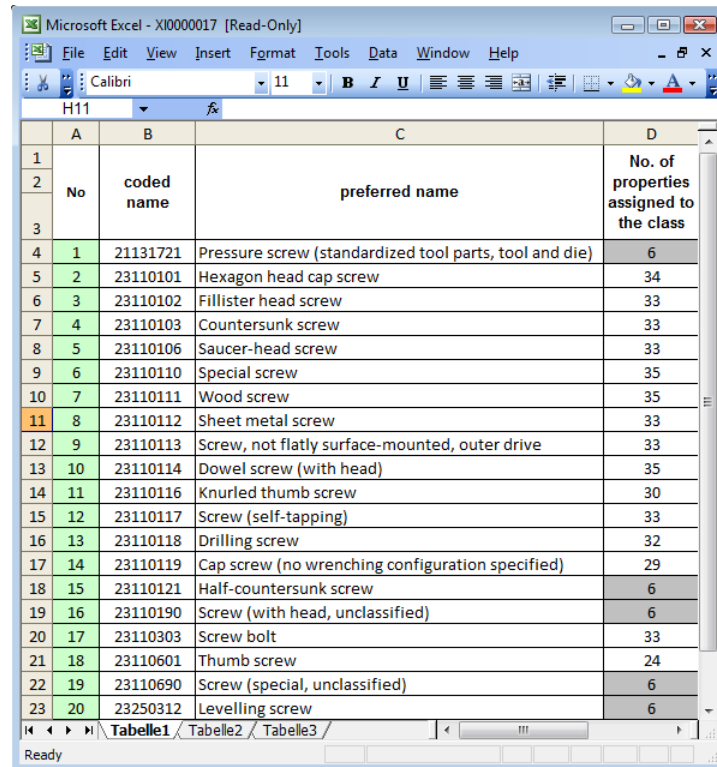
containing some common properties like product name, manufacturer name, etc. which can be factored out from the service and industry specific classes.

## Estimated reduction in the number of properties

With the introduction of the inheritance hierarchy for the application classes it is clear that the number of property-class assignments can be reduced. In the following we will calculate this reduction for the example of screws with the goal to give an estimation of the overall reduction of property-class assignments in eCI@ss.

In the case of screws we consider 34 classes, 20 of them are related to normal screws and 14 to medical screws.

Figure 2 shows the 20 classes related to normal screws and the total number of assigned properties to each class. Some of these classes are well developed. This means, that for example, 'Hexagon head cap screw' has got 34 properties, thus it is defined quite detailed in terms of associated properties.



	A	B	C	D
	No	coded name	preferred name	No. of properties assigned to the class
1				
2				
3				
4	1	21131721	Pressure screw (standardized tool parts, tool and die)	6
5	2	23110101	Hexagon head cap screw	34
6	3	23110102	Fillister head screw	33
7	4	23110103	Countersunk screw	33
8	5	23110106	Saucer-head screw	33
9	6	23110110	Special screw	35
10	7	23110111	Wood screw	35
11	8	23110112	Sheet metal screw	33
12	9	23110113	Screw, not flatly surface-mounted, outer drive	33
13	10	23110114	Dowel screw (with head)	35
14	11	23110116	Knurled thumb screw	30
15	12	23110117	Screw (self-tapping)	33
16	13	23110118	Drilling screw	32
17	14	23110119	Cap screw (no wrenching configuration specified)	29
18	15	23110121	Half-countersunk screw	6
19	16	23110190	Screw (with head, unclassified)	6
20	17	23110303	Screw bolt	33
21	18	23110601	Thumb screw	24
22	19	23110690	Screw (special, unclassified)	6
23	20	23250312	Levelling screw	6

Figure 2: Total number of properties assigned to 'normal screw classes'

On the other hand there are some classes that have only a few properties that are not sufficient to distinguish them from other classes. For example, the classes 'Pressure screw', 'Half-countersunk screw', 'Screw (with head unclassified)', and 'Levelling screw' have got only six properties. Five of them are basic properties and the additional one is the 'EAN code'.

Beside these basic properties there exist further properties which are often assigned to many classes. Figure 3 and Figure 4 show the set of properties assigned to the commodity classes representing the “normal screws”. Analysing these sets of properties, we can see that there are overlapping properties in the well developed classes. Or looking at this situation from the viewpoint of a class, some classes share common properties with other classes and a few classes have a number of specific properties like, for example, ‘Dowel screw’, ‘Screw bolt’ and ‘Thumb Screw’. From the total 48 screw properties there are 11 properties which are assigned only to one of the three mentioned classes (marked green in Figures 3 and 4 and collected in figure 5).

The screenshot shows a Microsoft Excel spreadsheet titled "Microsoft Excel - X10000028 [Read-Only]". The spreadsheet contains a table with 11 columns and 48 rows of data. The columns are labeled A through L, and the rows are numbered 1 through 48. The table lists various properties of normal screws and their assignment status across different classes.

No	Property set	13-17-	11-01-	11-01-	11-01-	11-01-	11-01-	11-01-	11-01-	11-01-	11-01-
1	BAA001003 - Manufacturer name	x	x	x	x	x	x	x	x	x	x
2	BAA002002 - Product type desc	x	x	x	x	x	x	x	x	x	x
3	BAA053003 - Supplier product n	x	x	x	x	x	x	x	x	x	x
4	BAD647002 - Manufacturer prod	x	x	x	x	x	x	x	x	x	x
5	BAA316003 - Product name	x	x	x	x	x	x	x	x	x	x
6	BAA271003 - EAN code	x	x	x	x	x	x	x	x	x	x
7	BAA833001 - Order supplement code	x	x	x	x	x	x	x	x	x	x
8	BAA900002 - Order supplement accor	x	x	x	x	x	x	x	x	x	x
9	BAB010001 - Publication date (year-mo	x	x	x	x	x	x	x	x	x	x
10	BAB165002 - standard letter to the sta	x	x	x	x	x	x	x	x	x	x
11	BAB637002 - Product class	x	x	x	x	x	x	x	x	x	x
12	BAE162001 - Requirement in accordanc	x	x	x	x	x	x	x	x	x	x
13	BAA838002 - Drive quantity	x	x	x	x	x	x	x	x	x	x
14	BAA923002 - Product class in accord	x	x	x	x	x	x	x	x	x	x
15	BAA932002 - Key width	x	x	x	x	x	x	x	x	x	x
16	BAA936001 - Tolerance information in	x	x	x	x	x	x	x	x	x	x
17	BAA937001 - Thread length	x	x	x	x	x	x	x	x	x	x
18	BAB072002 - Tolerance	x	x	x	x	x	x	x	x	x	x
19	BAB101002 - Surface protection	x	x	x	x	x	x	x	x	x	x
20	BAB112001 - Material in accordance wit	x	x	x	x	x	x	x	x	x	x
21	BAB150001 - Surface protection in acco	x	x	x	x	x	x	x	x	x	x
22	BAB664004 - Material	x	x	x	x	x	x	x	x	x	x
23	BAA901001 - Thread size of screwed plug										
24	BAA902001 - Thread size of nut end										
25	BAA905002 - Thread diameter threaded end										
26	BAA907001 - Thread diameter	x	x	x	x	x	x	x	x	x	x
27	BAA908002 - Thread direction	x	x	x	x	x	x	x	x	x	x
28	BAA909003 - Thread pitch at nut end	x	x	x	x	x	x	x	x	x	x
29	BAA910002 - Design of insertion end										
30	BAA914001 - Width of screw head										
31	BAA916001 - Head diameter of screw	x	x	x	x	x	x	x	x	x	x
32	BAA917002 - Head form	x	x	x	x	x	x	x	x	x	x
33	BAA918001 - Position of thread coating	x	x	x	x	x	x	x	x	x	x
34	BAA919001 - Screw length	x	x	x	x	x	x	x	x	x	x
35	BAA920001 - Length of threaded end										
36	BAA921001 - Length of nut end										
37	BAA922001 - Length of thread coating	x	x	x	x	x	x	x	x	x	x
38	BAA923001 - Length of thread coating at screw end										
39	BAA924001 - Length of thread coating at nut end										
40	BAA930001 - Shaft diameter of the screw										
41	BAA935001 - Width of ligament										
42	BAB090001 - Head form pursuant to st	x	x								
43	BAB093001 - Head length of screw										
44	BAB162001 - Height of head	x	x	x	x	x	x	x	x	x	x
45	BAB341001 - Thread size	x	x	x	x	x	x	x	x	x	x
46	BAB618001 - Thread design according	x	x	x	x	x	x	x	x	x	x
47	BAB166002 - Max. perm. lower deviation of nominal diameter										
48	BAE452001 - Max. distance to D1										
Grand Total		6	34	33	33	33	35	35	33	33	35

Figure 3: Properties assigned to 'Normal Screw', Part 1

No	Property set	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-	23-11-
1	BAA001003 - Manufacturer name	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
2	BAA002002 - Product type description	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
3	BAA003003 - Supplier product name	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
4	BAD047002 - Manufacturer product name	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
5	BAA316003 - Product name	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
6	BAA271003 - EAN code	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
7	BAA839001 - Order supplement	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
8	BAA300002 - Order supplement	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
9	BAB010001 - Publication date (year)	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
10	BAB165002 - Standard letter to technical drawing	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
11	BAB637002 - Product class	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
12	BAE162001 - Requirement in accordance with standard	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
13	BAB637002 - Product class	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
14	BAE162001 - Requirement in accordance with standard	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
15	BAA839002 - Drive quantity	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
16	BAA323002 - Product class in accordance with standard	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
17	BAA332002 - Key width	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
18	BAA336001 - Tolerance information	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
19	BAA337001 - Thread length	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
20	BAB072002 - Tolerances	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
21	BAB101002 - Surface protection	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
22	BAB118001 - Material in accordance with standard	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
23	BAB150001 - Surface protection	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
24	BAB664004 - Material	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
25	BAA301001 - Thread size of screwed plug	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
26	BAA302001 - Thread size of nut end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
27	BAA305002 - Thread diameter threaded end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
28	BAA307001 - Thread diameter	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
29	BAA308002 - Thread direction	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
30	BAA309003 - Thread pitch at nut end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
31	BAA310002 - Design of insertion end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
32	BAA314001 - Width of screw head	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
33	BAA316001 - Head diameter of screw	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
34	BAA317002 - Head form	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
35	BAA318001 - Position of thread on head	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
36	BAA319001 - Screw length	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
37	BAA320001 - Length of threaded end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
38	BAA321001 - Length of nut end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
39	BAA322001 - Length of thread on head	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
40	BAA323001 - Length of thread coating at screw end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
41	BAA324001 - Length of thread coating at nut end	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
42	BAA330001 - Shaft diameter of the screw	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
43	BAA335001 - Width of ligament	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
44	BAB030001 - Head form pursuant to standard	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
45	BAB033001 - Head length of screw	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
46	BAB162001 - Height of head	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
47	BAB341001 - Thread size	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
48	BAB618001 - Thread design according to standard	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
49	BAB166002 - Max. perm. lower deviation of nominal diameter	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
50	BAE452001 - Max. distance to D1	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
51	Grand Total	30	33	32	29	6	6	33	24	6	6	33	24	6	6	33	24	6	6	33

Figure 4: Properties assigned to 'Normal Screw', Part 2

Figure 5 shows the 11 different unique properties assigned to one of the three classes.

1	<b>BAA901001 – Thread size of screwed plug</b>
2	<b>BAA902001 – Thread size of nut end</b>
3	<b>BAA905002 – Thread diameter threaded end</b>
4	<b>BAA910002 – Design of insertion end</b>
5	<b>BAA920001 – Length of threaded end</b>
6	<b>BAA921001 – Length of nut end</b>
7	<b>BAA923001 – Length of thread coating at screw end</b>
8	<b>BAA924001 – Length of thread coating at nut end</b>
9	<b>BAA935001 – Width of ligament</b>
10	<b>BAB166002 – Max perm. Lower deviation of nominal diameter</b>
11	<b>BAE452001 – Max. Distance to D1</b>

*Figure 5: Unique properties in normal Screw group*

The main issue is to reduce the number of properties in the overlapping classes (see Figure 3 and Figure 4) and move them from the lowest level (Specific properties for specific screw) to higher levels (General classes with general properties and General application class). The total number of properties cannot be reduced; but the number of property assignments to classes can be reduced.

The reduction of properties assigned to classes is achieved by application classes and property inheritance. In the following we assume that a well developed commodity class for normal screw should have about 36 properties. Then we assume that a commodity class of normal screw group has to contain at least 4 specific properties, 15 general properties for general screw and 17 properties of general application classes.

To calculate the total reduction of property-class assignments in the group normal screw we first compute the total number of properties which are needed in the old scheme without the hierarchy of application classes. From the figures in the tables above we can conclude that on the average there are four specific properties in a commodity class, i.e. four properties are defined at that level per class and not inherited from an upper class. In addition, at the general technical level we count 15 properties, and at the general level we count 17 properties. All of them have to be associated to any of the 20 classes. Thus, for the 20 classes we get 720 properties (20 classes multiplied by (4 properties – 1st level, 15 properties – 2nd level, 17 properties – 3rd level).

$$20 \times (17 + 15 + 4) = 720$$

On the other hand, with property inheritance for application classes, we need to assign the properties of the general and the general technical level only once to a class. Thus, we get the following formula:

$$20 \times 4 + 15 + 17 = 112$$

Thus, the total number of property assignments to classes is restricted to 112. From 720 assigned properties in the first model, 112 property assignments to classes remain. That means that only 15% of the property class assignments remain and we get a reduction of about 85%.

## **General estimations on the reduction of properties**

Based on these examples of calculations for specific groups, we can do estimation for the whole eCI@ss classification.

There are more than 30,000 commodity classes in eCI@ss. A well developed commodity class should have about 30 properties.

In the old eCI@ss approach this would result in about 1,000,000 assignments of individual properties to individual commodity classes.

If we assume that we can introduce general commodity classes (like general screw) that cover on average about 10 commodity classes, then we will have 3,000 general classes.

If we assume that 80% of the properties can be assigned to the general classes, and inherited by the commodity classes, the number of individual assignments of properties will go down from about 1,000,000 to about 200,000.

If we assume that 50% of the properties that could be assigned to general classes could be transferred to general properties, then 100,000 assignments can be managed by inheritance. The assignments of properties will go down to about 100,000

The total reduction of assigned properties could be reduced by roughly 90% according to this rough calculation.

Of course the various segments have different approaches in the development of properties. In the service segment there are many overlapping properties that could be removed from individual commodity classes. That could lead to much higher reduction of properties assigned to commodity classes.

## **CONCLUSIONS**

This chapter has discussed two topics:

- First, it gave an overview about the requirements for describing the semantics of product information in e-business exchange processes. We have described the modelling levels to be addressed, gave a brief description of PLIB as one important data model for classifications and introduced a number of important classifications which are in practical use in e-business exchange processes. Finally, we have illustrated a few aspects of the adoption of the PLIB model in the classification eCI@ss.
- Second, it was argued that there is need for more flexibility and multiple views on a classification to serve different requirements and to overcome problems of the classifications which are imposed by the product categorization hierarchies. Specifically, we have proposed to add a second class hierarchy to eCI@ss and illustrated a number of benefits which can be gained by such an extension.



To summarize, we expect that by introducing such parallel hierarchies in addition to the existing categorization hierarchies used in classifications like eCl@ss, UNSPSC or GPC a number of benefits can be achieved, including the following:

1. Less effort for handling properties and their relationships to classes
2. Improved consistency and completeness of classes by linking them to a generic class and thus inheriting automatically the generic properties and ensuring that similar products are described by similar properties.
3. Better quality of property definitions
4. Less mistakes in property definitions
5. Harmonisation of classes and properties.

The practical use of classifications is still far behind the expectations which were raised ten years ago. Some of the reasons for this are the following:

- Users see different classifications which exist in parallel.
- Many users complain that the existing classifications do not fulfill their requirements.
- The costs and additional resources for building classifications may be very high.

Some of the ideas of this chapter might provide a contribution to further work addressing the above mentioned issues:

### **Many parallel classifications**

Since many of the classifications organize their classification hierarchies without clear and checkable categorization criteria, it is no surprise that the different classifications are organised differently as a response to the needs of their actual users. Thus, a number of classifications have been developed for similar areas with different class structures. That provides additional burden for users if they need to support different classifications for the communication with different partners. Therefore, it might be desirable to combine several classification hierarchies also across classifications. For instance, is it possible to use the UNSPSC hierarchy together with the eCl@ss properties? Or can a branch specific classification be linked to a horizontal classification? Thus, questions like the following have to be investigated:

- Can the class structures be mapped across classifications?
- Can the class structures of different classifications be used in combination (i.e. can they be used as different indexes to the same commodity classes)
- Can categorization hierarchies be combined with other property definition classifications?

One key issue to be resolved is the abstraction levels of classes in different classifications. In the CEN Workshop CMAP (CEN, 2011), a mapping exercise between eCl@ss, GPC, UNSPSC and CPV, the classification of the EU, is being performed, and it will be interesting to see how far these classifications really can be mapped, whether the product groups which they represent by their classes are really on a comparable level, and whether and how the classification hierarchies of different classifications can be combined with properties and property based hierarchies.

### **Do classifications respond to the requirements of all users?**

Most of the existing classifications have their origin in sales and procurement. On this basis they have grown controlled by some active users. But do they really support the requirements of

various different users? For example, one difference between the viewpoints of buyers and manufacturers is the comparability of products: The desire of buyers is to have a standardized comparison of products of different manufacturers, whereas the goal of manufacturers is to highlight their unique selling proposition (USP) which might not fit to the comparison criteria of the buyer. But also the different phases in the lifecycle of a product require different information about the product, maybe different search criteria, different organisation structures for products and properties, etc. Here our approach shows one possible solution by providing different views and classification structures which might also be added for other purposes and other circumstances. Currently, a research project with the participation of eCI@ss has been initiated which is looking into the topic of different requirements which have to be fulfilled by a horizontal classification standard.

### **Automatic generation of classification structures**

Most classifications are developed and maintained by committees and working groups according to a definition of the maintenance process. This is also true for the classification hierarchy, and especially the hierarchy is a topic of many debates in these working groups – due to the implicit criteria used for building the hierarchy. If we now add another hierarchy, these working groups have another area of potential discussions and thus additional work. Even for the *is-a* relationship between classes there are many different ways to organise such a hierarchy.

On the other hand, it would be desirable to build the inheritance hierarchy with the help of tools which construct clusters of properties. Opportunities to build such tools might come from the current activities in software refactoring (see e.g. Mens & Tourwé, 2004 and Streckenbach & Snelting, 2004) and from the area of formal concept analysis (see e.g. Ganter, Stumme & Wille, 2005).

Definitely, such automatic generation will avoid manual work. But it might be a challenge to convince the developers and users of classification standards to deal with automatically generated structures. To do that, it is important that the generated structures show some stability even in case of modifications of the underlying classes and properties. Due to the use of classifications in some master data management systems of companies, these companies are very sensitive to changes in their hierarchies.

### **ACKNOWLEDGEMENTS**

The authors like to thank Hui Dong who investigated in her master thesis various areas of the eCI@ss classification which provided us with some background information. We are particularly grateful for the support and the patience which we received from the editors of this book during the revision phase. And we like to mention the very instructive and supportive comments of the unknown reviewers which helped us to considerably improve the content of this chapter.

### **REFERENCES**

CEN (2006): CWA 15556-1-2006: CEN Workshop Agreement (2006): *Product Description and Classification – Part 1: New Property Library*. Ref. No.: CWA 15556-1:2006. Retrieved August 7, 2011 from <ftp://ftp.cen.eu/CEN/Sectors/List/ICT/CWAs/CWA15556-01-2006-Mar.pdf>.

CEN (2010): CWA 16138: CEN Workshop Agreement: *Classification and catalogue systems used in electronic public and private procurement*. Ref. No.: CWA 16138-1:2010. Retrieved August 7, 2011 from <ftp://ftp.cen.eu/CEN/Sectors/List/ICT/CWAs/CWA16138.pdf>.

CEN (2011): *Final Revised Business Plan for CEN/ISSS WS eCAT*. Retrieved August 6, 2011 from <http://www.cen.eu/cen/Sectors/Sectors/ISSS/Workshops/Pages/eCAT.aspx>.

European Commission (2008): *Regulation (EC) No. 213/2008*. Retrieved August 7, 2011 from [http://simap.europa.eu/codes-and-nomenclatures/codes-cpv/codes-cpv\\_en.htm](http://simap.europa.eu/codes-and-nomenclatures/codes-cpv/codes-cpv_en.htm).

Ganter, B., Stumme, G., Wille, E. (Ed.). (2005): *Formal concept analysis: foundations and applications*. Berlin, Heidelberg: Springer

Hepp, M., Leukel, J., Schmitz, V. (2007): A Quantitative Analysis of Product Categorization Standards: Content, Coverage, and Maintenance of eCl@ss, UNSPSC, eOTD, and the RosettaNet Technical Dictionary. *Knowledge and Information Systems (KAIS)*, 1 (13), 77-114.

Hepp, M. (2006). Products and services ontologies: A methodology for deriving OWL ontologies from industrial categorization standards. *International Journal on Semantic Web & Information Systems*, 2(1), 72–99.

Hesselmann, K., Ondracek N. (2006): *Strategiepapier Neues Datenmodell eCl@ss (Strategic Paper on the new data model of eCl@ss)*. Version 1.5. Internal eCl@ss Paper (in German).

International Electro technical Commission [IEC] (IEC 2009a): *IEC 61360-1: Standard data element types with associated classification scheme for electric components - Part 1: Definitions - Principles and methods*.

International Electrotechnical Commission [IEC] (IEC 2009b): *IEC 61987-10: Industrial-process measurement and control - Data structures and elements in process equipment catalogues - Part 10: List of Properties (LOPs) for Industrial-Process Measurement and Control for Electronic Data Exchange – Fundamentals*.

International Standardization Organisation [ISO] (1994): *ISO 10303-1: Industrial automation systems and integration -- Product data representation and exchange -- Part 1: Overview and fundamental principles*.

International Standardization Organisation [ISO] (2003): *ISO 15926-2: Industrial automation systems and integration -- Integration of life-cycle data for process plants including oil and gas production facilities -- Part 2: Data model*.

International Standardization Organisation [ISO] (2004): *ISO 10303-1: Industrial automation systems and integration -- Product data representation and exchange -- Part 11: Description methods: The EXPRESS language reference manual*.

International Standardization Organisation [ISO] (2006): *ISO 13584-511: Industrial automation systems and integration -- Parts library -- Part 511: Mechanical systems and components for general use -- Reference dictionary for fasteners*.

International Standardization Organisation [ISO] (2007a): *ISO 12006-3: Building construction -- Organization of information about construction works -- Part 3: Framework for object-oriented information*.

International Standardization Organisation [ISO] (2007b): *ISO 15926-4: Industrial automation systems and integration -- Integration of life-cycle data for process plants including oil and gas production facilities -- Part 4: Initial reference data*.

International Standardization Organisation [ISO] (2009): *ISO 23584-1: Optics and photonics -- Specification of reference dictionary -- Part 1: General overview on organization and structure*.

International Standardization Organisation [ISO] (2010a): *ISO 13584-42 Ed2: Industrial automation systems and integration -- Parts library -- Part 42: Description methodology: Methodology for structuring part families*.

International Standardization Organisation [ISO] (2010b): *ISO 22745-1: Industrial automation systems and integration -- Open technical dictionaries and their application to catalogues -- Part 1: Overview and fundamental principles*.

Löffelmann G., Polke, B., Zgorzelski, P. (2007): PROLIST Lists of Properties – an important step toward an integrated electronic engineering and business workflow. *atp international* 5 (1), 34-50.

International User Association for Automation in Process Industries [NAMUR] (2010): *NE 100 Version 3.2: Use of Lists of Properties in Process Control Engineering Workflows*.

Mens, T., Tourwé, T. (2004): A Survey of Software Refactoring, *IEEE Transaction on Software Engineering*, 30(2), 126-139

Reusch, P., Garcia, E. (2008): On Entropies in the Classification of Commodities - a Challenge for E-Commerce. *Communication of the Scientific Advisory Board of eCI@ss*, No. 1.

Reusch, P., Garcia, E. (2009): Harmonization of classes and property sets in critical commodity classes of eCI@ss. *Communication of the Scientific Advisory Board of eCI@ss*, No. 2.

RosettaNet (2007): *RosettaNet Technical Dictionary (RNTD)*, Version 4.2. Retrieved August 8, 2001 from [www.rosettanet.org](http://www.rosettanet.org).

Streckenbach, M., Snelting, G. (2004): Refactoring Class Hierarchies with KABA. *19th Annual ACM Conference on Object-Oriented Programming, Systems, Languages, and Applications*.

## **KEY TERMS AND DEFINITIONS**

**Classification:** a set of concepts, basically classes, properties and relationships between them, with the purpose to allow the linkage of products to product groups (classes) and to describe products in exchange processes between different business partners by commonly defined properties.

**Classification data model:** data model providing the resources for defining classifications.

PLIB: a classification data model, published as ISO 13584-42 (2010 in edition 2)

eCl@ss: a classification standard which has adopted since 2011 PLIB as its underlying data model.

Property: element of a classification which can be used to describe the characteristics of products.

Categorization class: a specific type of class which does not contain any properties and may not take part in a *is-a* hierarchy.

Application class: specific class type defined in eCl@ss V7.0, which carries the properties and is linked to a single commodity class in the eCl@ss hierarchy.